NASA-TM-111491

7N-82-TM
48092

Weighing Hypotheses: Incremental Learning from Noisy Data

Philip Laird
Artificial Intelligence Research Branch
MS 269-2
NASA Ames Research Center
Moffett Field, CA 94035-1000

NASA Ames Research Center
Artificial Intelligence Research Branch

Technical Report FIA-93-01

January, 1993

Weighing Hypotheses: Incremental Learning from Noisy Data

Philip Laird[*]
AI Research Branch
NASA Ames Research Center
Moffett Field, California 94035-1000 (U.S.A.)

Abstract

Incremental learning from noisy data presents dual challenges: that of evaluating multiple hypotheses incrementally and that of distinguishing errors due to noise from errors due to faulty hypotheses. This problem is critical in such areas of machine learning as concept learning, inductive programming, and sequence prediction. I develop a general, quantitative method for weighing the merits of different hypotheses in light of their performance on possibly noisy data. The method is incremental, independent of the hypothesis space, and grounded in Bayesian probability.

Introduction

Consider an incremental learning system for which the input data may be noisy. As the system learns, it maintains a set of hypotheses. Each example or input datum may support some hypotheses and contradict others. As learning proceeds, new—and probably more complex—hypotheses may need to be introduced as the available ones prove to be unacceptable. The question we address in the paper is: *How do we quantify our relative confidence in the various hypotheses, while accounting for the possibility of noisy data and an expanding hypothesis space?*

If the input data are noise free, the question is easy: any contradiction between data and hypothesis eliminates the hypothesis. With noise, however, no hypothesis can be permanently discarded until the evidence of its unsuitability is sufficiently strong. Moreover we generally prefer simpler hypotheses to more complex ones, but there is often a tradeoff between the complexity of the hypothesis and the amount of error it suffers on the input data.

To clarify the issues, consider the following situation. The problem is to infer from a sequence of integers a rule in the form of a finite difference equation that predicts the remainder of the sequence. We stop looking as soon as we are "sufficiently confident" (i.e., are willing to bet) that we have the best rule. The system must, therefore, be incremental: following each input value, it must decide whether to stop or to continue and which hypotheses are viable explanations.

[*]LAIRD@PLUTO.ARC.NASA.GOV

Now suppose that we have obtain the first input value: $x_1 = 1$. Four of the many possible rules for this sequence from the meager evidence are:

- H_1 The sequence consists of all ones.
- H_2: The sequence begins with 1, and for all $n > 1$, $x_{n+1} = x_n + 1$.
- H_3: The sequence begins with 1, and for all $n > 1$, $x_{n+1} = x_n \times 2$.
- H_4: The sequence begins with 1, and for all $n > 1$, $x_{n+1} = (x_n - 1)^2 + 2$.

All these hypotheses agree with the first value $x_1 = 1$ of the series, and except for H_1 (which predicts 1 for the next value), all predict 2 for the second value. The first hypothesis requires no previous values in order to make its prediction ("one"), whereas the other three base their prediction for the next value on a single previous value in the sequence.

Suppose the second value x_2 is in fact 2. We cannot yet discard the first hypothesis because x_2 may be in error. But intuitively our confidence diminishes that H_1 will correctly predict x_3.

Next we learn that $x_3 = 3$. H_1 now has incorrectly predicted two values, H_3 has incorrectly predicted one value, and the others have correctly predicted two values. In view of our lack of faith in H_1, an algorithm might well decide to replace it with the hypothesis:

- H_5: The sequence begins with 1 and 2, and for all $n > 2$, $x_{n+2} = x_{n+1} + x_n$.

An incremental algorithm does not test new hypotheses on all previous input examples, but with our omniscience we can see that H_5 correctly predicts x_3 given the values x_1 and x_2. The fact that H_5 requires two prior values to predict the next instead of just one means that it will not begin making predictions until x_5, assuming that the algorithm has stored only x_3. On the other hand, H_5 is syntactically less complex than H_4, and perhaps H_3 as well if multiplication is an expensive operation.

It will appear in the AAAI Spring Symposium on Training Issues in Incremental Learning, March, 1993.

Next we learn that $x_4 = 5$. At this point we can summarize our knowledge about these five hypotheses:

Hypothesis Name	Correct Predictions	Incorrect Predictions	Relative Complexity
H_1	0	3	small
H_2	2	1	medium
H_3	1	2	medium
H_4	2	1	high
H_5	0	0	medium

Assume that we are seeking a single theory (hypothesis) to explain the data rather than a weighted combination of several hypotheses. On which hypothesis should we place our bets for predicting x_5, and how much confidence should we have in the result?

Again, the fundamental problem is to find an incremental technique to weigh the merits of hypotheses when the data are noisy and the constraint of incrementality implies, among other things, that hypotheses may not all be evaluated using the same data. Solving this problem in a general and principled way is the purpose of this paper. Any approach to this question is likely to be somewhat subjective, since we are formalizing the notion of preference; but any reasonable approach should at least make the nature of its subjectivity explicit, and if possible should allow different preferences within the same framework.

The theory to be described offers these features.

Background

The problem of weighing competing hypotheses enjoys a long history of debate among philosophers and statisticians. Given a family $\mathcal{H}$ of hypotheses and a set E of evidence, one has to select the best hypothesis from $\mathcal{H}$ and to quantify the likelihood $\Pr(H \mid E)$ that the hypothesis H in $\mathcal{H}$ accounts for E. Popper [Popper, 1972] studies the problem from the viewpoint of weighing competing scientific theories and develops a measure of confidence that credits a hypothesis most for correctly predicting a result that otherwise would be highly improbable. I. J. Good [Good, 1983], who has studied this problem for some time, proposes a measure of the weight W of evidence E in favor of the hypothesis H given by

$$W = \log \frac{\Pr(E \mid H)}{\Pr(E \mid \overline{H})},$$

where $\overline{H}$ is the logical negation of the hypothesis H.

Related to Good's measure are the information-theoretic measures that in effect credit a hypothesis with the information obtained from the input examples. For many domains one can construct a hypothesis that accounts for any finite number of examples (for example, an n'th order difference equation can account for any sequence of $n+1$ integers). Constantly replacing a hypothesis that accounts for all n examples so far with a more complex one that accounts for $n+1$ examples is simple memorization, not learning: the hypothesis must achieve a certain amount of compression of the input data [Board and Pitt, 1990]. The term *minimum description length* (MDL) model is often used when one's confidence in a hypothesis is proportional to the expected amount of compression the hypothesis achieves in explaining the example data. In the case of noisy data or an incomplete hypothesis class, we can compress the input data by writing down a hypothesis, omitting all the input examples correctly determined by the hypothesis, and listing only the examples that conflict with the hypothesis. A better hypothesis has fewer exceptions, but it must balance accuracy against syntactic complexity in order that the total size of the hypothesis and the data be minimal.

Classical hypothesis testing is another approach. Here one evaluates each hypothesis individually, comparing its ability to predict the data to that of a null hypothesis that makes random guesses for its predictions.

Some scholars, including Good, argue for a universal measure, a "best" measure of the weight of evidence in a hypothesis. Except, perhaps, for the evaluation of scientific theories, this author is not convinced by these arguments and believes that practical matters such as computational efficiency usually play a role in judging hypotheses. Incrementality is one such efficiency constraint: the algorithm must revise its confidence measure following each new input, without the need to save and reprocess more than a small, bounded number of inputs. Bayesian models accomplish this if the sucessive input values are independent [Laird, 1989]; otherwise, computing the posterior distribution is not incremental. As a practical matter Bayesians typically assume that the inputs are independent, even when they clearly are not, as a "first-order" approximation to the true posterior. This kind of compromise occurs frequently when one must trade cogitation time for online availability of the inferences.

In the realm of machine learning, the problem of weighing competing hypotheses occurs whenever multiple inductive inferences are possible. Recently the need for a principled weight-of-evidence measures has arisen in the subfield of inductive logic programming [Quinlan, 1990; Muggleton and Feng, 1992]. Starting from a large collection of examples and counterexamples of a relation, ILP programs search for generalizations of the examples that explain as many of the positive examples and as few of the negative examples as possible. Since many generalizations are possible, a "least" generalization is sought, but computational complexity and other issues make prohibitive computing a unique least generalization. The need arises, therefore, to quantify the relative confidence in different generalizations. So far, MDL and non-Bayesian confirmation theory have been suggested [Gillies, 1992; Muggleton, 1988; Muggleton *et al.*, 1992], but theoretical and computational efficiency problems persist, and no fully satisfactory approach has been found.

Machine learning and computational learning theory contribute two important notions to our theory. The first is the importance of prediction. Any concept-learning algorithm[1] can be converted to a prediction algorithm of roughly the same complexity, and vice versa [Natarajan, 1991]. The idea is that a concept-learning algorithm can apply the concept it learns to predict the class of each instance from the attributes, and a prediction strategy can be turned into a concept by forming a list of the members of each class. The quality of a predictor is measured by the number of prediction mistakes it makes as a function of the number of examples it has seen. A polynomial-time learning algorithm (in the PAC sense) is characterized by a harmonic decline in the rate of prediction errors ($\mathcal{O}(1/n)$) relative to the number n of examples seen. In this paper I cast the theory in terms of a predictive learner, but by this equivalence the theory applies equally to concept learning.

The other notion is that of learning from noisy data, and in the PAC model of learning quite a lot is known. Here we summarize these results only briefly, but [Angluin and Laird, 1987; Laird, 1988; Sakakibara, 1991] can be consulted for more detail. The first task is to separate the process supplying the (correct) example data from the transmission channel through which the data values must pass. That channel may be noisy, and as a result the data presented to the learner may be incorrect.The *noise rate* is the fraction of input bits that differ from those actually transmitted by the source. In the worst case, systematic "adversarial" noise—in which the errors always work to the maximum disadvantage of the learning algorithm—may favor bad hypotheses over good ones and thereby make it impossible to select a good hypothesis unless the noise rate is quite small.

A less-than-worst-case scenario, however, assumes that the noise is more random than systematic, and since random errors in the input affect both good and bad hypotheses alike, bad hypotheses will, with high likelihood, still perform worse than good ones. Consequently a learning algorithm that searches for a hypothesis in perfect agreement with the input data can be changed to accommodate noise merely by choosing a least inconsistent hypothesis, i.e., one that on average disagrees least with the input examples. The cost incurred by the need to account for noise is that more hypotheses must be entertained until the number of input examples is statistically large enough to support the inference that some hypotheses are inferior. The details of the statistical procedure depend on the noise model one is assuming.

[1]Concept learning is the task of learning from examples to assign a class name to objects based on their attributes. An example is learning to identify birds from their plumage, song, etc.

In this paper I have resisted selecting a single noise model, just as I have elected not to limit the theory to a single domain or representation. Instead I assume merely that *the noise rate is sufficiently low, and the noise process sufficiently random, that on average better hypotheses make fewer prediction mistakes than worse ones.* The emphasis is on a reasonably general solution to the problem of incremental learning from noisy data in preference to an *ad hoc* solution that applies to one representation and one noise model. By seeking such generality one is more likely to gain insight into the fundamental nature of the problem, with the result that designing effective algorithms for specific cases is easier and more systematic.

Modeling Predictive Failures

We adopt the following scenario: Input values x_1, $x_2, \ldots$ are presented to a learning algorithm whose task is to predict x_{n+1} from what it has learned on the preceding n values. Predictions must be made by selecting one of a countable family $\mathcal{H} = \{H_1, H_2, \ldots\}$ of hypotheses and using it to predict the next value.[2]

The algorithm selects possible hypotheses from $\mathcal{H}$, but at any time it is considering only finitely many of them. The set of candidate hypotheses it considers may vary over time as unsuccessful ones are discarded and replaced by new ones. The algorithm is incremental in that, if a new hypothesis H is added to the set of competing hypotheses before the arrival of the i'th value, the evaluation of H begins with the i'th value and continues with subsequent values, until it is discarded. Thus at any time the hypotheses under consideration may be supported or refuted by different examples.

To account for the fact that some hypotheses require fewer previous values than others (and thereby achieve greater compression), we assume that each hypothesis H has a fixed "latency" $\lambda_H \geq 0$. A latency of zero means that the hypothesis requires no previous input bits in order to predict the next (e.g., H_1 in the introductory example). A hypothesis with latency λ requires, on average, λ bits of input in order to construct a prediction for one bit. The value λ_H does not enter the theory explicitly, but it affects the algorithm implicitly in two ways:

1. The algorithm must remember as many bits as required by the candidate hypothesis with the largest latency. For this reason, an incremental algorithm may place an upper bound on the latency of hypotheses that it is willing to consider.

[2]Bayes-optimal algorithms form a prediction by combining predictions from all hypotheses in $\mathcal{H}$ and weighting each prediction by the prior probability of the hypothesis. In most circumstances this is computationally impractical, and often the family of acceptable hypotheses is not closed under linear combinations (e.g., sentences in first-order logic and finite-state automata).

2. After being introduced, a new hypothesis must await the input of λ bits before it can be credited with predicting any bits correctly. This may reduce its effectiveness relative to other hypotheses introduced simultaneously but requiring fewer input bits before they can begin predicting.

Suppose we introduce a new hypothesis H chosen arbitrarily. What kind of rate of prediction success do we expect from H? Viewing the input stream as a sequence of bits, with some values having more bits than others, we expect that H will predict some bits correctly and others incorrectly. Its success or failure in predicting each bit is a random variable that, in general, is governed by a complex stochastic process. But in the absense of other information, we may reasonably model this prediction process as follows:

> *A randomly selected hypothesis H has a finite probability p_H of incorrectly predicting any bit b of the input stream X. The (unknown) value p_H is fixed for H and independent of b. Thus the likelihood of correctly predicting an input value x of length x bits is $(1-p_H)^{|x|}$.*

According to this model, the hypothesis H is correct iff $p_H = 0$; if H is incorrect, the likelihood of a run of correct predictions decreases exponentially with the length of the run.

To forestall any misunderstanding, note that we are *not requiring* this property to be true of our hypothesis space: It is no more than a simple binomial model of what is doubtless a complex process governing the pattern of prediction errors from an incorrect hypothesis. This assumption is not unlike that made by statisticians when they assume that the performance of a hypothesis on successive examples is statistically independent: the independence assumption makes at least approximate calculations feasible where otherwise no calculations at all would be tractable.

Note also that we are *not* requiring our hypotheses to predict the input stream one bit at a time: we simply measure prediction success in units of the number of bits correctly predicted. This is because input values differ in size, and we gain more confidence when a hypothesis correctly predicts a large input value x_i than when it predicts a short value. Let us assume that the input values and predictions are both encoded in some finite alphabet and that we can count the number of bits correctly predicted.

Confidence and Reliability

Noise-free case. Assume for the moment that the input values are noise free, and suppose that a hypothesis H has correctly predicted the past n bits worth of input examples beginning with input x_{k+1}. According to our model, if the hypothesis is faulty, the probability of its predicting all n of these bits correctly is $(1-p_H)^n$. The value of p_H is unknown, but we can estimate the probability density for it as follows. (For readability we shall omit the subscript H from p_H.)

Begin with the non-informative prior density of $f_0(p) = 1$ (for $0 \le p \le 1$). The probability of correctly predicting the $n+1$'st bit, given correct predictions on the past n bits, is:

$$\begin{aligned}\Pr(n+1 \mid n) &= \frac{\int_0^1 (1-p)^{n+1} dp}{\int_0^1 (1-p)^n dp} \\ &= \frac{n+1}{n+2}. \qquad (1)\end{aligned}$$

This formula we recognize as the Laplace Law of Succession.

Now we update the posterior density $f_n(p)$ for p after H has predicted n bits correctly, and for this we need our assumption that the accuracies of successive predictions are independent random variables. An application of Bayes's rule gives, for $n \ge 0$:

$$\begin{aligned}f_n(p) &= \frac{(1-p)^n}{\int_0^1 (1-\xi)^n d\xi} \\ &= (n+1)(1-p)^n. \qquad (2)\end{aligned}$$

As n increases, this density rapidly concentrates toward low error probabilities (Figure 1).

At this point we have obtained expressions for the probability that H will predict correctly the first (and each subsequent) bit of the next input example and a distribution for the parameter p. Next we compute a measure of how *confident* we are in the hypothesis H based on its track record of predictive accuracy. Confidence reflects what is at stake: if risk is low, I may be willing to bet on a hypothesis whose distribution $f_n(p)$ is not so concentrated at zero. One way to model this tradeoff is to introduce a parameter δ that measures the acceptable risk of endorsing an incorrect hypothesis. For 90% confidence, δ is taken to be 0.1; for 95% confidence, $\delta = 0.05$, etc. The numerical figure of merit we assign to H is a function of both the distribution $f_n(p)$ (which reflects the performance of the hypothesis) and the risk parameter δ.

So assume we have chosen a value $0 < \delta \le 1/2$. After H has predicted n bits, what estimate $\hat{p}$ for p can we adopt and be confident that, with probability at least $1-\delta$, $p \le \hat{p}$? We call $1-\hat{p}$ the *δ-reliability* of the hypothesis H and estimate its value as a function of n. By definition,

$$\int_0^{\hat{p}} f_n(p)dp = 1 - \delta.$$

Substituting (2) and solving for $\hat{p}$:

$$\hat{p}(n) = 1 - \delta^{1/(n+1)}. \qquad (3)$$

Note that $\hat{p}(n)$ is $1-\delta$ (i.e., high) for $n = 0$ and decreases exponentially to zero with increasing n.

Recalling the example above with the five sequence-extrapolation hypotheses, only H_5 is a viable hypothesis after seeing x_4. Assume it predicts correctly the

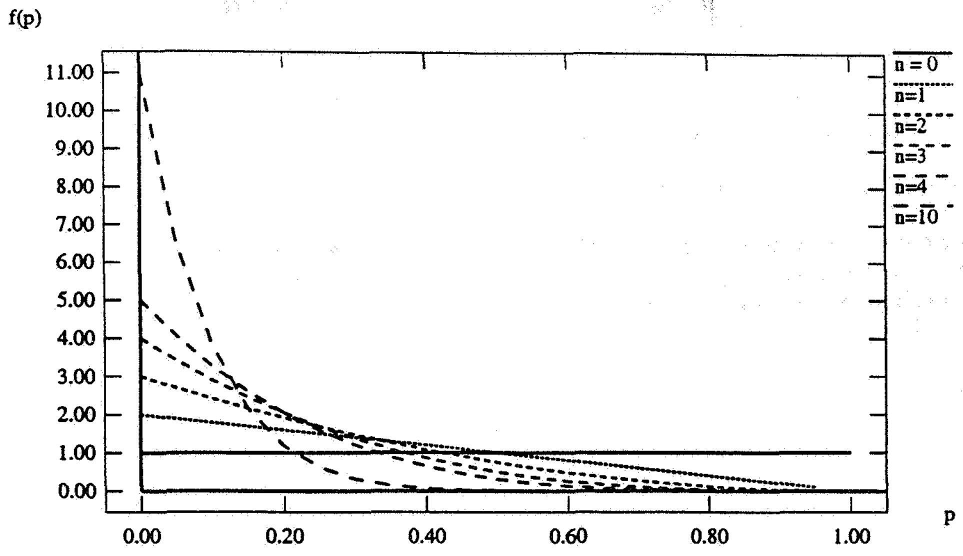

Figure 1: Graph of $f_n(p)$ for several values of n.

fifth input value $x_5 = 8$. Taking $|x_5| = 4$ bits and $\delta = 0.05$, we find that H_5 is expected to predict correctly the next bit with probability 5/6, and with 95% confidence we are willing to bet that H_5's probability of making an error on a random bit of the next example is less than $\hat{p}_{H_5} = 0.451$.

Suppose two hypotheses share identical records of success in predicting the input data, but differ in their syntax. According to this model both have the same reliability: the only feature that matters insofar as reliability is concerned is their track record in predicting the future. If a choice must be made, clearly we are free to choose the simpler hypothesis or the one that is computationally faster for making predictions. This syntactic criterion, however, applies only *after* the reliability criterion has been applied—a fact that distinguishes our model from most MDL approaches.

Worth emphasizing is that this model of confidence is *independent of the set of possible hypotheses.* We have not, for example, defined a prior distribution on the set of all possible hypotheses and computed posteriors based on the prediction success of each hypothesis. While this approach is useful in Bayesian hypothesis testing, it runs into difficulty when an incremental algorithm (like ours) introduces new hypotheses on the fly as old ones are discarded. Moreover it requires that syntactically variant but semantically equivalent hypotheses be recognized and handled jointly as a single hypothesis; frequently, however, the problem of deciding the equivalence of two hypotheses is undecidable or at least computationally intractable. In our model each hypothesis is evaluated without regard to what other hypotheses may be under consideration.

Noisy case. Now we consider the more interesting case: when the input values may be noisy. We must revise our calculation of the confidence and reliability of a hypothesis to account for noise. The math is somewhat more complicated, but the ideas are essentially the same.

Suppose that the hypothesis has correctly predicted all but r bits out of the past n. The likelihood of its correctly predicting the $n+1$'st bit is given by

$$\begin{aligned} \Pr(n+1 \mid n, r) &= \frac{\int_0^1 p^r(1-p)^{n-r+1}dp}{\int_0^1 p^r(1-p)^{n-r}dp} \\ &= \frac{n-r+1}{n+2}, \text{ for } 0 \le r \le n. \quad (4) \end{aligned}$$

When $r = 0$, this agrees with the previous noise-free result (1).

The posterior density $f_{n,r}(p)$ of p, the likelihood that the hypothesis will make an error predicting the next bit, can once again be calculated using Bayes's rule.

$$\begin{aligned} f_{n,r}(p) &= \frac{p^r(1-p)^{n-r}}{\int_0^1 \xi^r(1-\xi)^{n-r}d\xi} \\ &= (n+1)b_n(r \mid p), \qquad (5) \end{aligned}$$

where $b_n(r \mid p)$ is the binomial density. To solve for $1-\hat{p}$, the δ-reliability, we set

$$1 - \delta = \int_0^{\hat{p}} f_{n,r}(p)dp,$$

and after a bit of calculus we obtain

$$\delta = (1-\hat{p})^{n-r+1} \sum_{i=0}^{r} C_i \hat{p}^i, \qquad (6)$$

where

$$C_i = \begin{cases} 1 & (i=0) \\ \frac{(n-r+1)(n-r+2)\ldots(n-r+i)}{i!} & (i>0) \end{cases}$$

Again, note that Eqn. (6) reduces to the noise-free result (3) when $r = 0$.

Although we can't solve this expression explicitly for $1-\hat{p}$, when δ is given the sum (6) evaluates to a polynomial equation that we can solve numerically by standard methods. (It looks much more complicated than it really is).

Returning once more to the example, H_4 incorrectly predicts x_4 and x_5. Of the first $n = 12$ input bits, therefore, it incurs $r = 7$ bits of error. Thus its probability is about $(12-7+1)/14 = 0.43$ of correctly predicting the next bit, and with 95% confidence the calculation (6) estimates its minimum probability $\hat{p}_{H_4}$ of a prediction error as about 0.776. For the hapless H_1, $n = r = 12$, and it has only one chance in 14 of correctly guessing the next bit. Its minimum error probability $\hat{p}_{H_1}$ is virtually one at the 95% confidence level.

Factoring Confidence into the Learning Algorithm

How can we incorporate the preceding calculations into an incremental algorithm for learning from noisy data? We give an algorithm schema for incremental learning based on our weight-of-evidence model. The schema depends upon the specification for the following elements:

- A representation language for the input values ("examples") and a length measure over the symbols of that language. We call the units of measurement "bits" and denote the length of the example x by $|x|$.
- A family $\mathcal{H}$ of hypotheses, together with an efficient algorithm for deciding whether the hypothesis $H \in \mathcal{H}$ predicts the value x. Predictions from each hypothesis are expressed in the same language as used for the examples.
- An operatorΔ (the "divergence") that compares an input value x to a predicted value y and expresses the difference $x\Delta y$ in bits. The quantity $x\Delta y$ indicates the number of bits in the input x incorrectly predicted by the hypothesis; thus the divergence function must have the property: $0 \le (x\Delta y) \le |x|$.
- A schedule for introducing new hypotheses from $\mathcal{H}$ into a finite set of active candidate hypotheses.
- A criterion for deciding whether an active hypothesis should be retained or discarded, based upon its record of predictive success. One possibility is to eliminate hypotheses H for which $\hat{p}_H$ rises above some threshold at a given confidence level.
- A syntactic criterion that selects a preferred hypothesis from among a set of hypotheses that so far are equally successful in predicting input values.

As discussed above, we shall not assume any properties of the noise model except that on average good hypotheses make fewer prediction mistakes than worse ones.

Given these elements, we may construct the following learning algorithm: Let *ACTIVE* be a finite set of active hypotheses from $\mathcal{H}$, initially empty. Each hypothesis $H \in$ *ACTIVE* has a record of n_H bits predicted, r_H bits of which were incorrect.

```
1. Add to ACTIVE a set of fresh candidates from
   $\mathcal{H}$, setting $n = r = 0$ for each.
2. Get the next input example $x$.
3. For each $H \in$ ACTIVE:
3.1 Compute $x\Delta y$, where $y$ is $H$'s prediction for
    the input.
3.2 Set $n_H := n_H + |x|$ and $r_H := r_H + (x\Delta y)$.
3.3 If $H$ fails the retention criteria, discard $H$ from
    ACTIVE.
4. Let $\hat{p} = \max\{\hat{p}_H \mid H \in ACTIVE\}$
   and PREFERRED-HYPOTHESIS= the preferred hypothesis among the set
   $\{H \mid H \in ACTIVE, \hat{p}_H = \hat{p}\}$. Output the prediction of
   PREFERRED-HYPOTHESIS for the next example.
```

Naturally any reasoning about this algorithm, including its correctness and its complexity, will depend upon the above elements and upon the nature of the particular noise model.

An Application

As a simple, if somewhat contrived, illustration of how one might apply the theory to construct a learning algorithm, imagine that a power company is seeking to develop a usage profile for its largest customers in order to anticipate demand. The day is divided into 1440 minutes. A hypothesis for each customer is a continuous, piecewise linear curve $U(t)$, for $0 \le t \le 1439$, representing the predicted usage at time n. Such a function can be efficiently represented by a list of pairs $[t, U(t)]$, meaning that, between the times t and $t+1$ minutes, power usage is predicted to vary linearly between $U(t)$ units and $U(t+1)$ units.

The measured usage, of course, is a complex function with a lot of noise, and surely not piecewise linear. But we are seeking an approximation that captures peaks and valleys during the day. A sophisticated Bayesian or regression technique could be used to construct such a function, but these are not incremental methods and are needlessly complex since they look for an optimal, rather than acceptable, solution.

A meter samples the customer's actual power consumption once per minute. "Noise" in the form of random events cause a certain amount of daily variation in each customer's usage at each time. Seasonal effects also affect the variation, but for simplicity we assume that all variations other than diurnal ones have been eliminated.

The learning task is to learn a set of $[n, U(n)]$ pairs, numbering from one to 1440 pairs. A hypothesis with fewer than 1440 pairs makes predictions by linear interpolation at times not explicitly listed in the set. Such a hypothesis is equivalent to a hypothesis with all 1440 pairs in the set but is syntactically less complex. Since there is no difference in the predictions made by equivalent hypotheses, the size of the hypothesis plays no role in evaluating its predictive accuracy. The algorithm may, however, prefer to output or store a simpler hypothesis, and the schedule by which hypotheses are considered may introduce ones with fewer points first.

Suppose a particular hypothesis predicts a usage of U at time t. Let α be a fixed positive fraction. If the measured usage at time t is $\bar{U}(t)$, and $|U(t) - \bar{U}(t)| < \alpha|U(t)|$, then the prediction is credited with one "bit" of accuracy; otherwise it is charged an error of one bit. No credit is received for measured usage before the hypothesis is introduced.

The incremental learning schema we defined in a previous section requires a schedule for introducing candidate hypotheses. Presumably less detailed hypotheses (ones with fewer ranges) should be considered first, so hypotheses will be introduced in increasing order of the number of pairs, starting with one pair. For each pair there are two degrees of freedom: we can specify the time t and the predicted power level U. Although we could search systematically through this space for an optimal hypothesis, a more efficient approach is to use some simple statistics to estimate the expected times of day when the customer reaches the maximum and minimum usage levels and then to adjust these estimates by hillclimbing. Subsequent hypotheses with more ranges can be introduced by partitioning some of the ranges of a hypothesis into two separate ranges. (For purposes of this example we are not concerned with the detailed design of the algorithm). As hypotheses are introduced, evaluated, and discarded, no more than a few input values (times and readings) need be retained in memory in order to locate maxima and minima and to weigh the evidence in favor of each hypothesis.

Each active hypothesis is assigned a score representing the number of "bits" predicted correctly. Our confidence in each hypothesis is measured by the δ-reliability and computed using equation (6) above. The algorithm can be halted as soon as some hypothesis achieves a desired level of accuracy with sufficient confidence.

Regression and other statistical time-series techniques arguably may do a better job on this problem, but the example illustrates how an incremental learning algorithm can be developed in a systematic way for a problem where the data are noisy and the hypotheses vary in both complexity and detail.

Conclusions

Incremental learning, by its nature, entails dynamic reassessment of hypotheses in light of new evidence. We have defined a simple and useful mathematical theory based on prediction that quantifies the weight of evidence in favor of a hypothesis. Distinctive features of our theory include the following:

- The mathematics is based upon a simple, easily computed model of the way an incorrect hypothesis makes prediction errors and a generic model of how noise affects the rate of prediction errors.
- No effort is made to distinguish prediction errors due to noise from prediction errors due to a faulty hypothesis. This frees the theory from explicit dependence on a particular noise model and enables us to write down a generic incremental algorithm applicable to many problems. In particular, as $r \rightarrow 0$ in the equations, the noisy case reduces continuously to the noise-free case.
- The quality of a hypothesis is measured independently of the space of competing hypotheses—i.e., each hypothesis is evaluated in terms of its own performance record, regardless of what other hypotheses might be competing with it. By contrast with conventional Bayesian hypothesis-testing methods, we can easily introduce and discard hypotheses on the fly without having to reinitialize all our statistical measurements.
- The MDL preference for smaller hypotheses is sharpened by separating those syntactic elements that enable the hypothesis to predict more values from those that have no bearing at all on the way the hypothesis predicts the input data.

Finally, the algorithm schema for incremental learning from noisy data applies to many kinds of learning—including concept learning and sequence prediction—where the performance of a hypothesis can be tested by its ability to predict features of the input data. Such a schema serves as a template for building learning algorithms for a variety of domains, and like version spaces [Mitchell, 1978] and refinement algorithms [Laird, 1988], such schemas obviate reinventing an algorithm for every new representation by factoring out the syntax-independent features of the procedure.

Acknowledgments

Ronald Saul reviewed a draft of this paper and corrected several errors.

References

D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2:343–370, 1987.

R. Board and L. Pitt. On the necessity of Occam algorithms. In *Proc. 22nd ACM Symposium on Theory of Computing*, pages 54–63, 1990.

D. Gillies. Confirmation theory and machine learning. In *International Workshop on Inductive Logic Programming*, 1992. (unpublished).

I.J. Good. Weight of evidence: a brief survey. In *Bayesian Statistics 2*, pages 249–270. North Holland, 1983.

P. Laird. *Learning from Good and Bad Data*. Kluwer Academic, 1988.

P. Laird. A survey of computational learning theory. In R. Banerji, editor, *A Sourcebook on Formal Techniques in Artificial Intelligence*. Springer Verlag, 1989.

P. Laird. Discrete sequence prediction and its applications. In *Proc., 9th National Conference on Artificial Intelligence*. AAAI, 1992.

T. M. Mitchell. *Version Spaces: an approach to concept learning*. PhD thesis, Stanford University, 1978.

S. Muggleton and C. Feng. Efficient induction of logic programs. In *Inductive Logic Programming*, pages 281–298. Academic Press, 1992.

S. Muggleton, A Srinivasan, and M. Bain. Compression, significance, and accuracy. In *Machine Learning: Proceedings of the Ninth International Workshop*. Morgan Kaufmann, 1992.

S. Muggleton. A strategy for constructing new predicates in first-order logic. In *Proc. 3rd European Working Session on Learning*. Pitman, 1988.

B. Natarajan. *Machine Learning: A Theoretical Approach*. Morgan Kaufmann, 1991.

K. Popper. *The Logic of Scientific Discovery*. Hutchinson, 1972.

R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

Y. Sakakibara. *Algorithmic learning of formal languages and decision trees*. PhD thesis, Tokyo Institute of Technology, 1991. (Also available as Research Report IIAS-RR-91-22E, International Institute for Advanced Study of Social Information Science, Fujitsu Laboratories Ltd., Numazu, Shizuoka JAPAN 410-03).